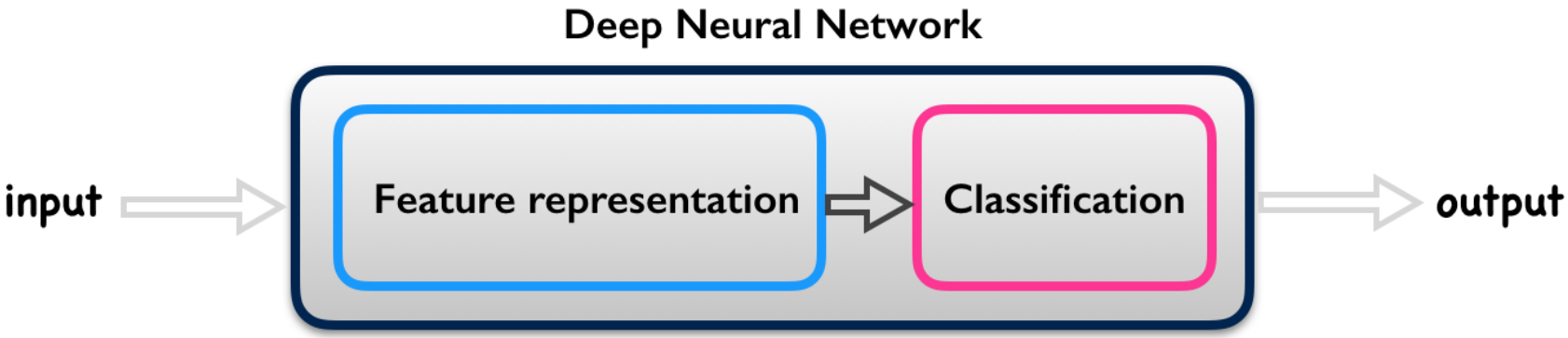
Convolutional Matching Methods

Zaid Harchaoui

March 29, 2016

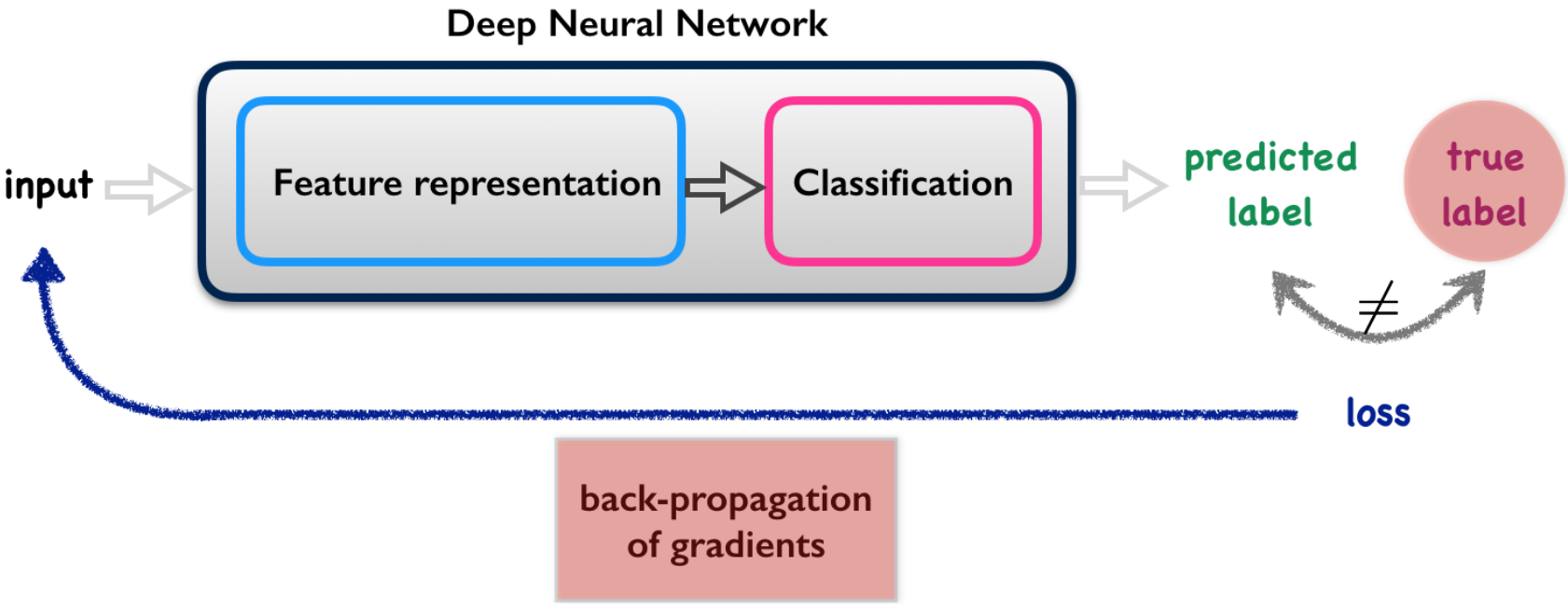
Overview of Deep Neural Networks

Overview of Deep Neural Networks



Training of Deep Neural Networks

Training of Deep Neural Networks



Deep Learning approach

Current methodology

1. Frame the task as predicting output **label** from input **example**
2. Collect a **huge training sample**
3. Train using **supervised learning** and **stochastic back-prop**
4. Done

Deep Learning approach

Current methodology

1. Frame the task as predicting output **label** from input **example**
2. Collect a **huge training sample**
3. Train using **supervised learning** and **stochastic back-prop**
4. Done

Challenges

1. Can any task be framed as a prediction task?
2. Where do I get the huge training sample?
3. Training with stochastic back-prop, is it that easy?

Framing the task as prediction

Image retrieval

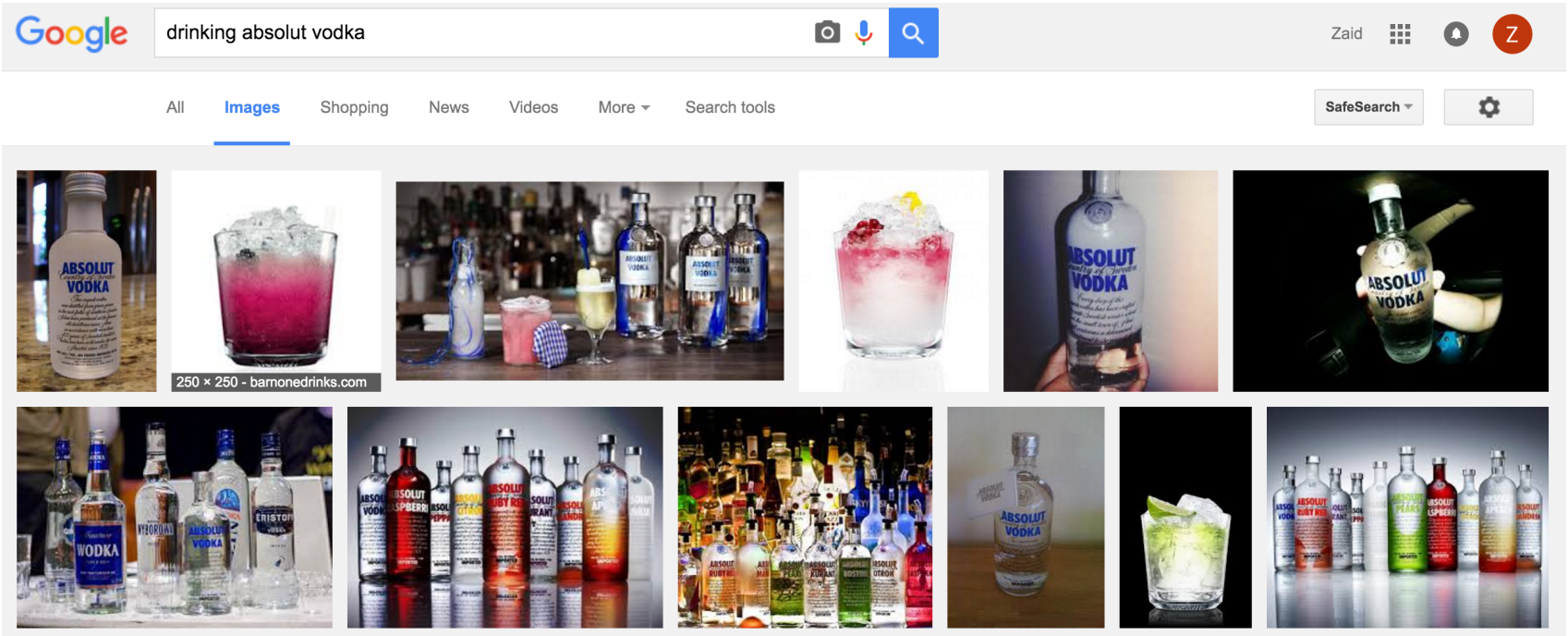


Figure: Google image search results for query "drinking absolut vodka".

Framing the task as prediction

Limitations of reframing

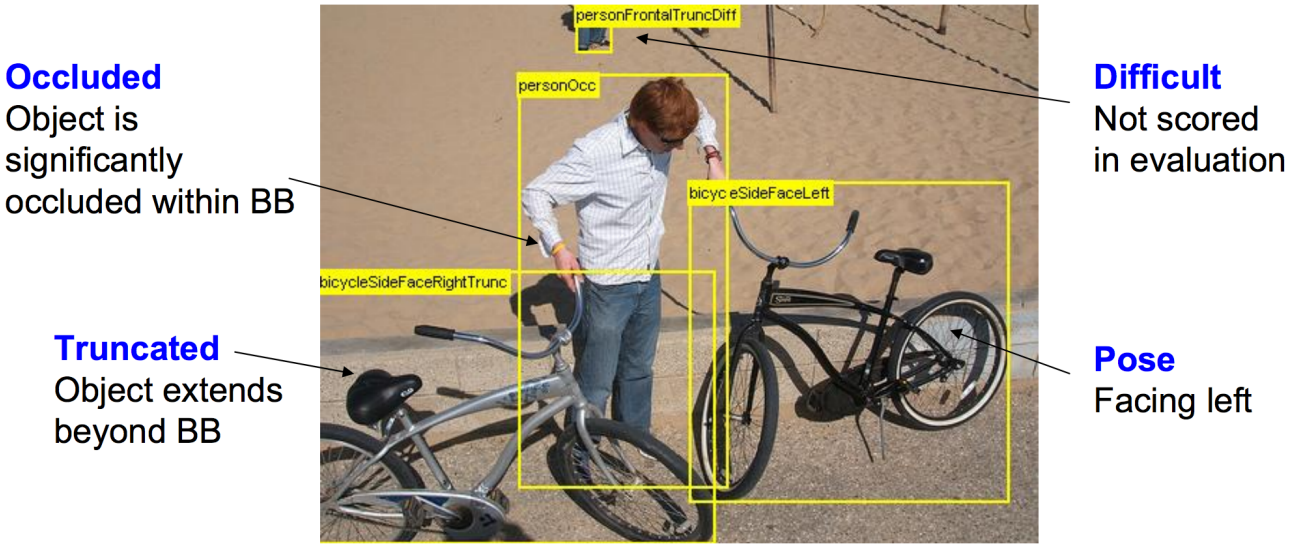
Computer vision is not a statistical problem

Car examples in ImageNet

Is this less of a car
because the context is wrong?

Figure: From Leon Bottou's keynote at ICML 2015.

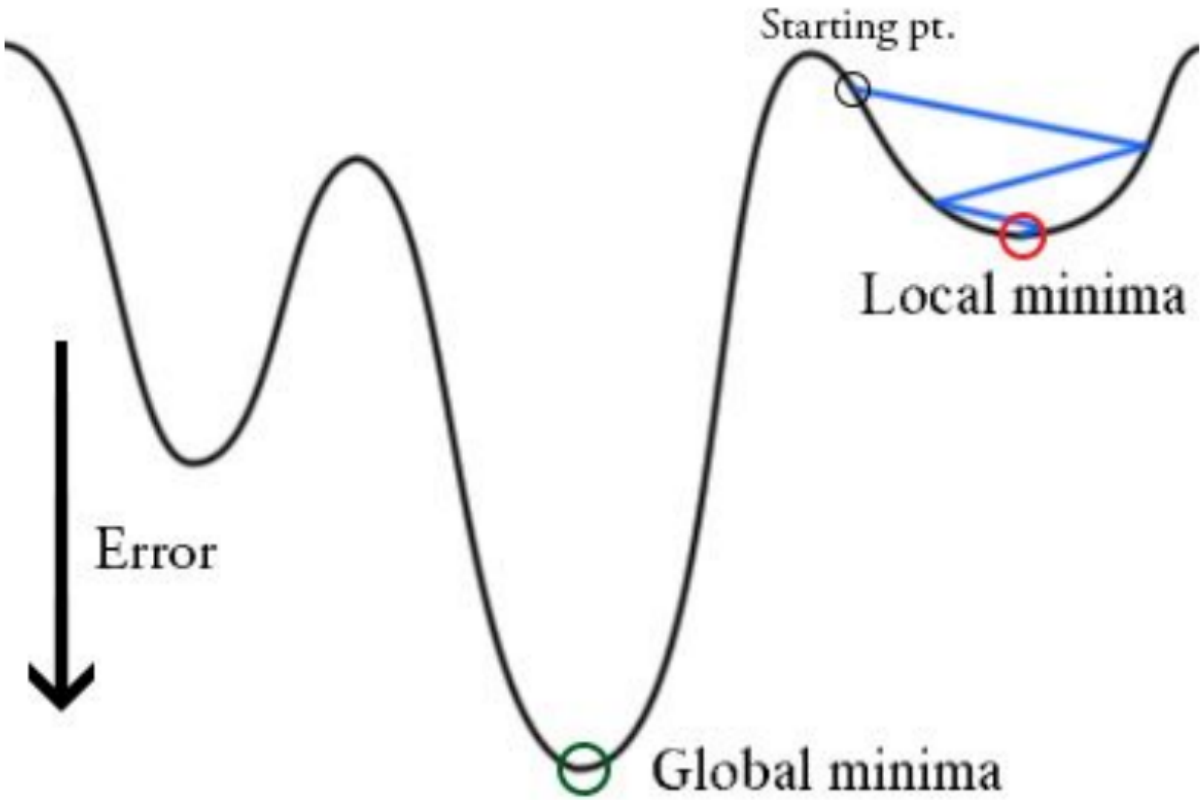
Collecting huge labelled training sample



Getting reliable human annotations is:

- time-consuming
- expensive
- often ambiguous

Training Deep Nets with back-prop



The wall of supervision

Current methodology

1. Frame the task as predicting output label from input example
2. Collect a huge training sample
3. Train using supervised learning and stochastic back-prop
4. Done

For some application, current methodology, which relies on **supervision**, hits a wall

Convolutional Matching Methods

Research program

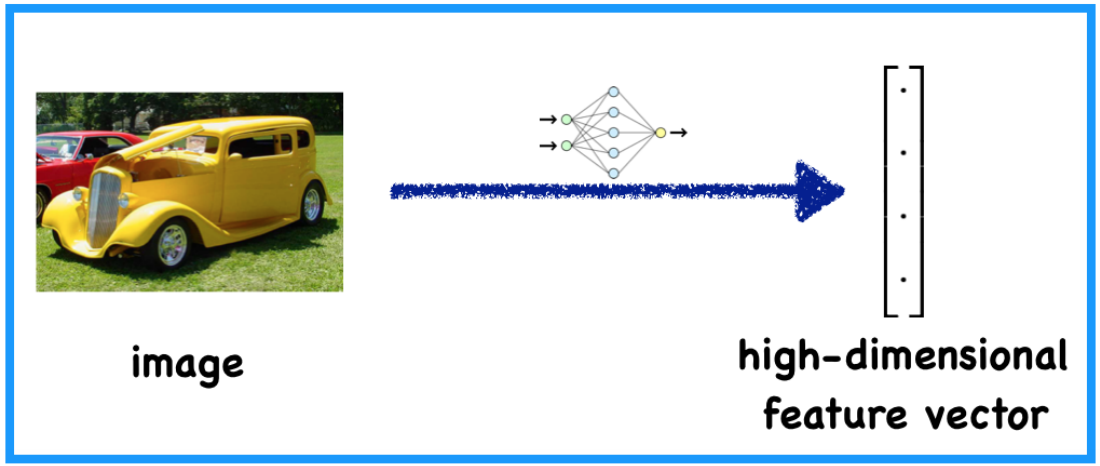
1. Clear and simple mathematical design principle
2. Training with little or no supervision
3. Layer-by-layer training using stochastic gradient optimization

Real-world applications

- Image retrieval
- Motion estimation (optical flow)

Feature representation of Deep Neural Nets

Feature representation



Design	Neuroscience-inspired
Property	Local invariance
Aesthetic	Depth

Convolutional kernels

M M'

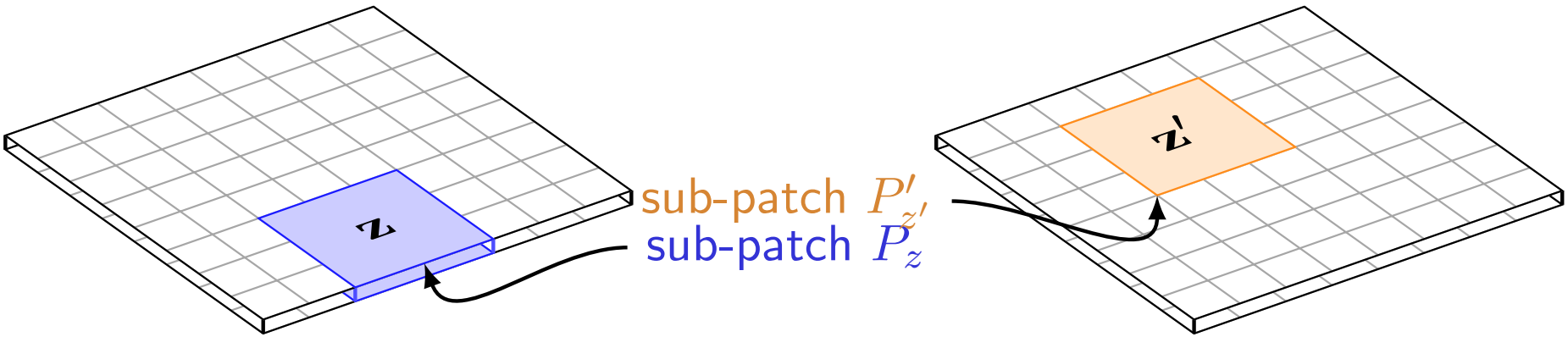
similarity measure

"kernel"

Design	**Theoretically-grounded**
Property	Local invariance
Aesthetic	**Conciseness**

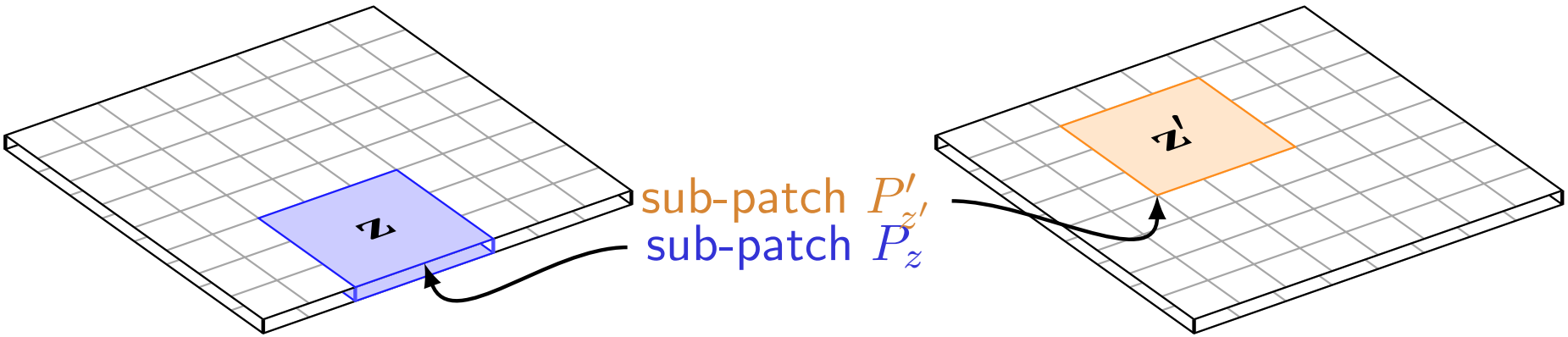
Kernel between sub-patches

$$\kappa(P, P') = \|P\|\|P'\|e^{-\frac{1}{2\alpha^2}\|\tilde{P}-\tilde{P}'\|^2}$$



Kernel between images

$$K(M, M') = \sum_{z,z' \in \Omega} e^{-\frac{1}{2\beta^2}\|z-z'\|^2} \underbrace{\|P\|\|P'\| e^{-\frac{1}{2\alpha^2}\|\tilde{P}-\tilde{P}'\|^2}}_{\kappa\left(P_z, P'_{z'}\right)}$$



Convolutional kernel between images

Single-layer convolutional kernel

$$K(M, M') = \sum_{z,z' \in \Omega} e^{-\frac{1}{2\beta^2}\|z-z'\|^2} \underbrace{\|P\|\|P'\| e^{-\frac{1}{2\alpha^2}\|\tilde{P}-\tilde{P}'\|^2}}_{\kappa\left(P_z, P'_{z'}\right)}$$

Main components

- **Shift-invariance** thanks to kernel $\exp\left(-\frac{1}{2\beta^2}\|z - z'\|^2\right)$
- **Matching patches** through kernel κ
- **Permutation-invariance** thanks to sum over all locations $\sum_{z,z' \in \Omega}$

Convolutional kernel between images

Single-layer convolutional kernel

$$K(M, M') = \sum_{z,z' \in \Omega} \underbrace{e^{-\frac{1}{2\beta^2}\|z-z'\|^2}}_{\text{kernel bw positions}} \underbrace{\|P\|\|P'\| e^{-\frac{1}{2\alpha^2}\|\tilde{P}-\tilde{P}'\|^2}}_{\text{kernel bw sub-patches}}$$

Main components

- **Shift-invariance** thanks to kernel $\exp\left(-\frac{1}{2\beta^2}\|z - z'\|^2\right)$
- **Matching patches** through kernel κ
- **Permutation-invariance** thanks to sum over all locations $\sum_{z,z' \in \Omega}$

Multi-layer convolutional kernel

Multi-layer convolutional kernel

Kernel comparing patches from φ_M^{k-1} and $\varphi_{M'}^{k-1}$

$$\sum_{u,u' \in \Omega_{k-1}} e^{-\frac{1}{2\beta_k^2}\|u-u'\|^2} \kappa_k(\varphi_M^{k-1}(u), \varphi_M^{k-1}(u'))$$

where

$$\kappa_k(\varphi, \varphi') = \|\varphi\|_{\mathcal{H}_{k-1}} \|\varphi'\|_{\mathcal{H}_{k-1}} e^{-\frac{1}{2\alpha_k^2}\|\varphi-\varphi'\|^2_{\mathcal{H}_{k-1}}}$$

What we need

- compact approximation of φ_M to propagate through layers
- efficient scheme to recursively compute kernel

Multi-layer convolutional kernel

Multi-layer convolutional kernel

Kernel comparing patches from φ_M^{k-1} and $\varphi_{M'}^{k-1}$

$$\sum_{u,u' \in \Omega_{k-1}} e^{-\frac{1}{2\beta_k^2}\|u-u'\|^2} \kappa_k(\varphi_M^{k-1}(u), \varphi_M^{k-1}(u'))$$

where

$$\kappa_k(\varphi, \varphi') = \|\varphi\|_{\mathcal{H}_{k-1}} \|\varphi'\|_{\mathcal{H}_{k-1}} \underbrace{e^{-\frac{1}{2\alpha_k^2}\|\tilde{\varphi}-\tilde{\varphi}'\|^2_{\mathcal{H}_{k-1}}}}_{\text{expensive to compute!}}$$

What we need

- compact approximation of φ_M to propagate through layers
- efficient scheme to recursively compute kernel

Numerical approximation of the kernel
Convolutional Kernel Net

Step 1: explicit embedding of each layer feature representation

$$\sum_{u,u' \in \mathcal{P}_k} e^{-\frac{1}{2\beta_k^2}\|u-u'\|^2} \kappa_k(\varphi_M^{k-1}(u), \varphi_M^{k-1}(u'))$$
$$\approx \sum_{u,u' \in \mathcal{P}_k} e^{-\frac{1}{2\beta_k^2}\|u-u'\|^2} \tilde{M}_k(u)^T \tilde{M}_k(u)$$

Step 2: uniform sampling approximation of Gaussian kernel

$$\sum_{u,u' \in \mathcal{P}_k} e^{-\frac{1}{2\beta_k^2}\|u-u'\|^2} \tilde{M}_k(u)^T \tilde{M}_k(u')$$
$$\approx \frac{2}{\pi} \sum_{v \in \Omega_k} \left(\sum_{u \in \mathcal{P}_k} e^{-\frac{1}{2\beta_k^2}\|u-v\|^2} \tilde{M}_k(u) \right)^T \left(\sum_{u' \in \mathcal{P}_k} e^{-\frac{1}{2\beta_k^2}\|u'-v\|^2} \tilde{M}_k(u') \right)$$

Key idea

Finite-dimensional approximation (aka explicit embedding)

For all $\mathbf{x}$ and $\mathbf{y}$ in $\mathbb{R}^q$,

$$e^{-\frac{1}{2\alpha^2}\|\mathbf{x}-\mathbf{y}\|_2^2} \approx \sum_{j=1}^{p} f_j(\mathbf{x}) f_j(\mathbf{y})$$

Division of Gaussian kernel in the convolution sense

We have the relation

$$e^{-\frac{1}{2\alpha^2}\|\mathbf{x}-\mathbf{y}\|_2^2} = \left(\frac{2}{\pi\alpha^2}\right)^{\frac{q}{2}} \int_{\mathbf{z}\in\mathbb{R}^q} e^{-\frac{1}{\alpha^2}\|\mathbf{x}-\mathbf{z}\|_2^2} e^{-\frac{1}{\alpha^2}\|\mathbf{y}-\mathbf{z}\|_2^2} d\mathbf{z}$$

Key idea

Division of Gaussian kernel in the convolution sense

We have the relation

$$e^{-\frac{1}{2\alpha^2}\|\mathbf{x}-\mathbf{y}\|_2^2} = \left(\frac{2}{\pi\alpha^2}\right)^{\frac{q}{2}} \int_{\mathbf{z}\in\mathbb{R}^q} e^{-\frac{1}{\alpha^2}\|\mathbf{x}-\mathbf{z}\|_2^2} e^{-\frac{1}{\alpha^2}\|\mathbf{y}-\mathbf{z}\|_2^2} d\mathbf{z}$$

Possible strategies

- Monte-Carlo approximation $\rightarrow$ random Fourier features
- Integral quadrature $\rightarrow$ Nyström approximation, kernel herding
- **Direct optimization**

Key idea

Factorization of Gaussian kernel in the convolution sense

We have the relation

$$e^{-\frac{1}{2\alpha^2}\|\mathbf{x}-\mathbf{y}\|_2^2} = \left(\frac{2}{\pi\alpha^2}\right)^{\frac{q}{2}} \int_{\mathbf{z}\in\mathbb{R}^q} e^{-\frac{1}{\alpha^2}\|\mathbf{x}-\mathbf{z}\|_2^2} e^{-\frac{1}{\alpha^2}\|\mathbf{y}-\mathbf{z}\|_2^2} d\mathbf{z}$$

Direct Optimization

1. Initialize with convex relaxation
2. Minimize using **Catalyst**-accelerated randomized incremental gradient

$$\min_{W,b} \quad \mathbb{E}_{x,x'\sim\mathbb{P}_X} \left[e^{\frac{\|x-x'\|^2}{2\alpha^2}} - \sum_{j=1}^{p} e^{w_j^\top x + b_j} e^{w_j^\top x' + b_j} \right]^2$$

Convolutional Kernet Nets

Overview

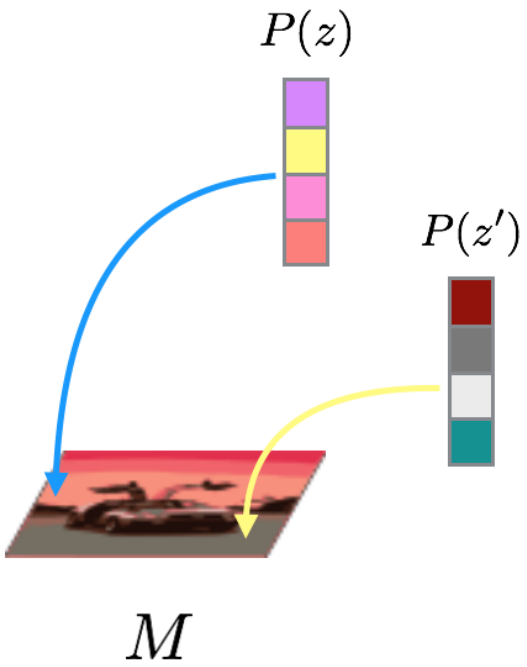
Convolution $W_k^T P_k(z)$

Exponential Nonlinearity $\tilde{M}_k(z) = \|P_k(z)\| \exp\left(W_k^T P_k(z) + b_k\right)$

Gaussian Pooling $M_k(z) = \sum_{u \in \Omega_{k-1}} \exp\left(-\frac{1}{\beta_k^2}\|u - z\|^2\right) \tilde{M}_k(u)$

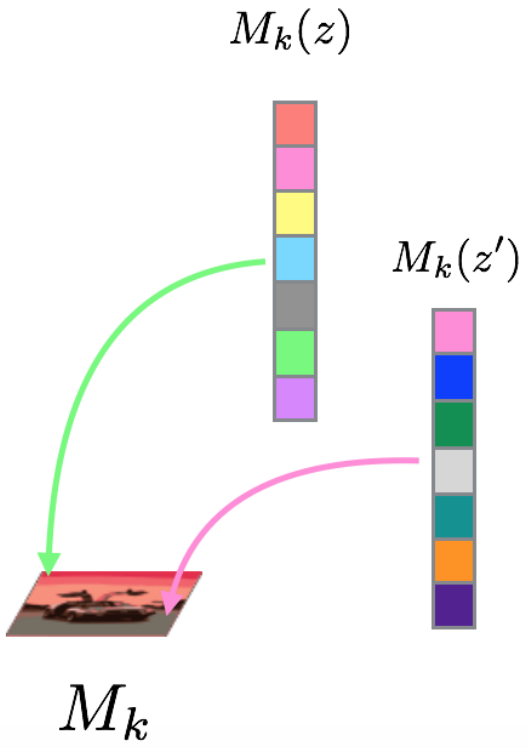
Convolutional Kernet Nets

Zoom on zero-th layer



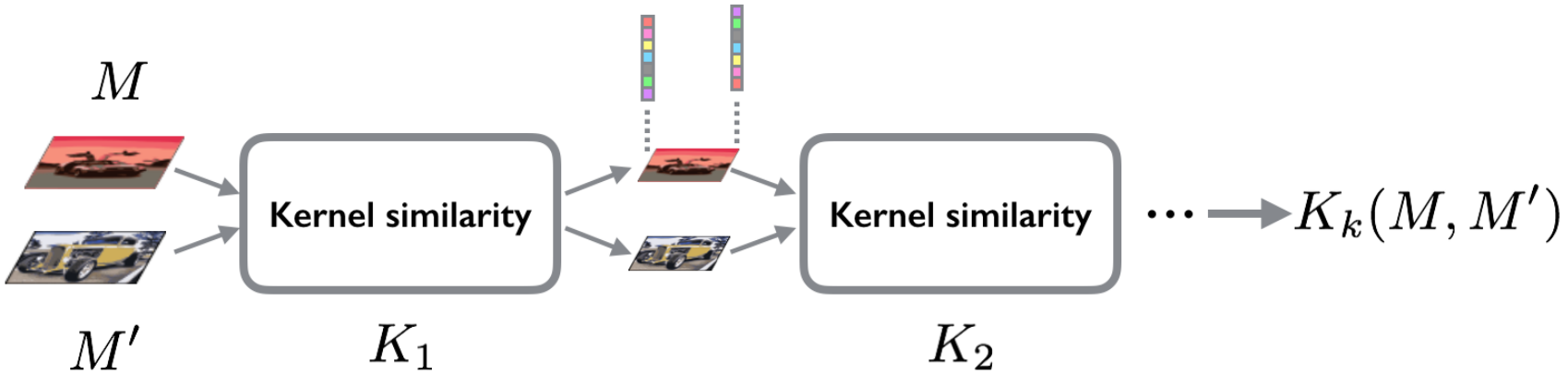
Convolutional Kernet Nets

Zoom on k-th layer



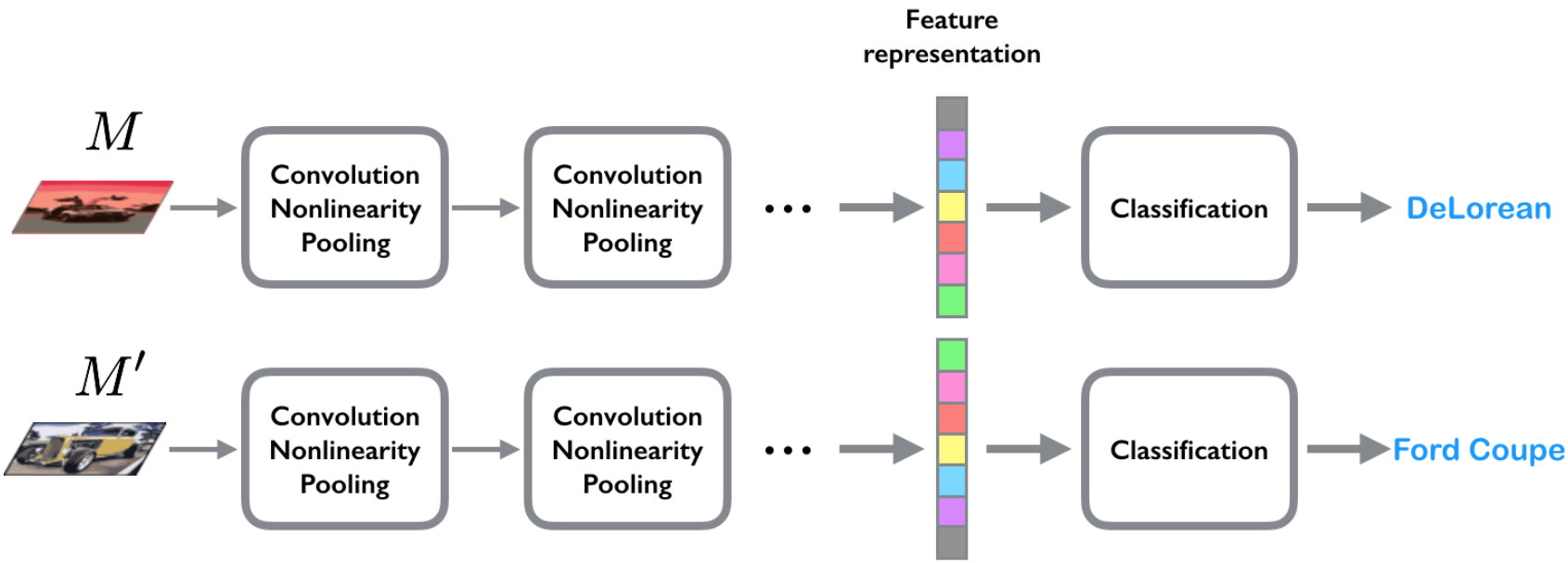
Convolutional Kernet Nets vs Convolutional Neural Nets

Convolutional Kernel Nets



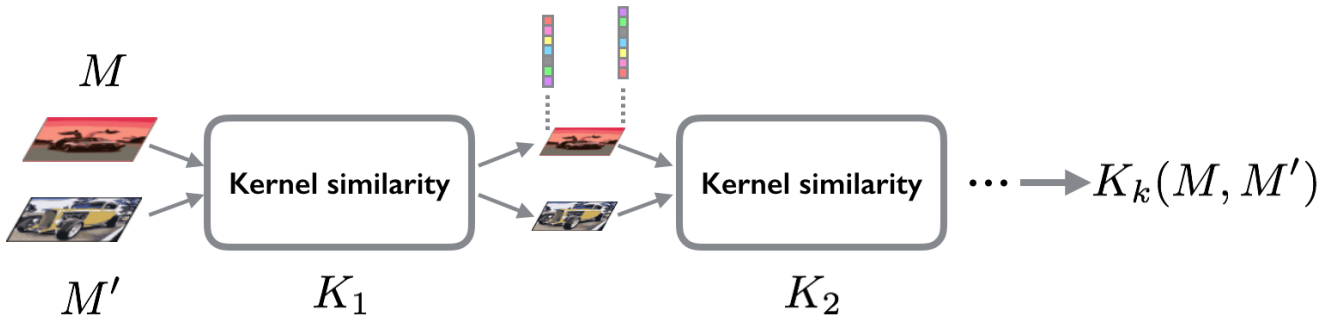
Convolutional Kernet Nets vs Convolutional Neural Nets

Convolutional Neural Nets



Convolutional Kernet Nets vs Convolutional Neural Nets

Convolutional Kernel Nets



Convolutional Neural Nets

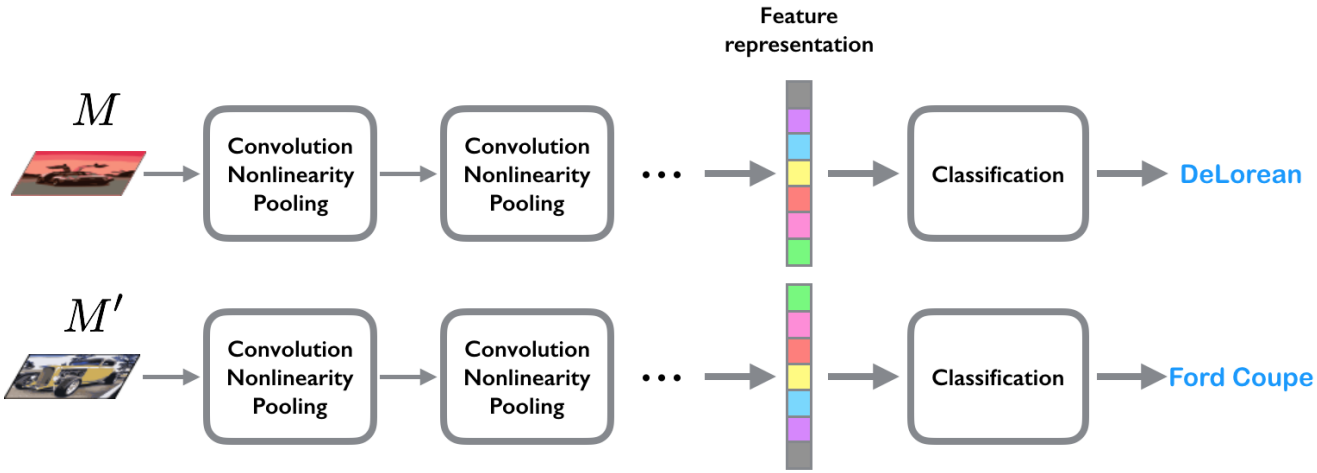


Image retrieval

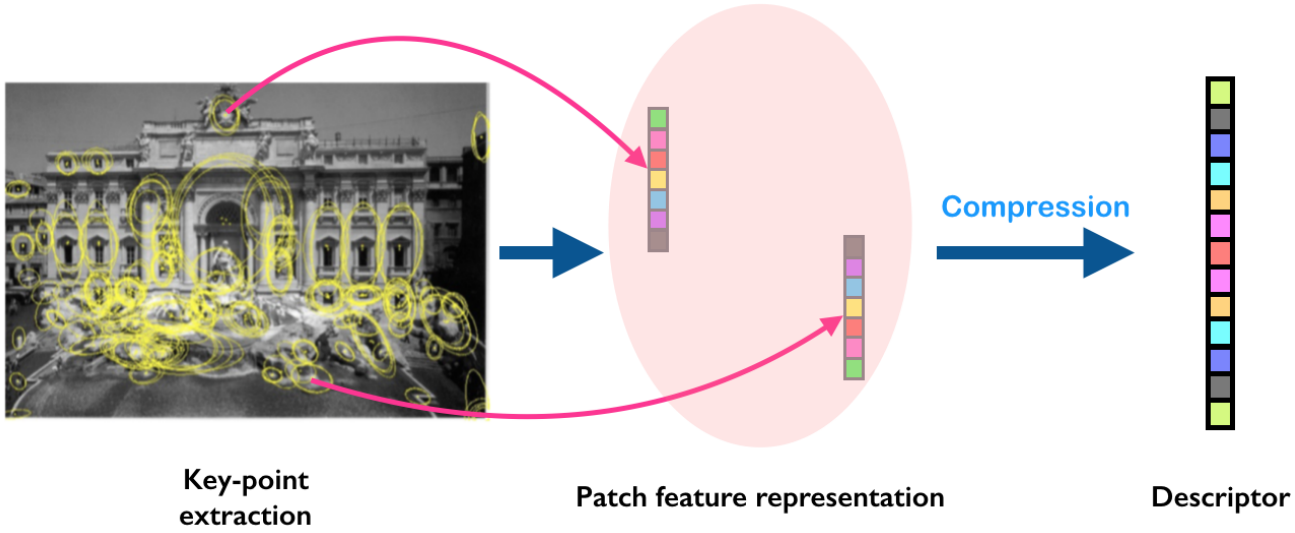


Figure: Image retrieval pipeline

Experiments: extracting keypoints

Figure: Example of matching points in two different images. Salient points are extracted (left), affine rectified (middle), and normalized in rotation (right). Note that the resulting level of invariance that remains to be covered by the patch descriptor is relatively low, as most of the work has been accomplished by the detector.

Input cues to CKN

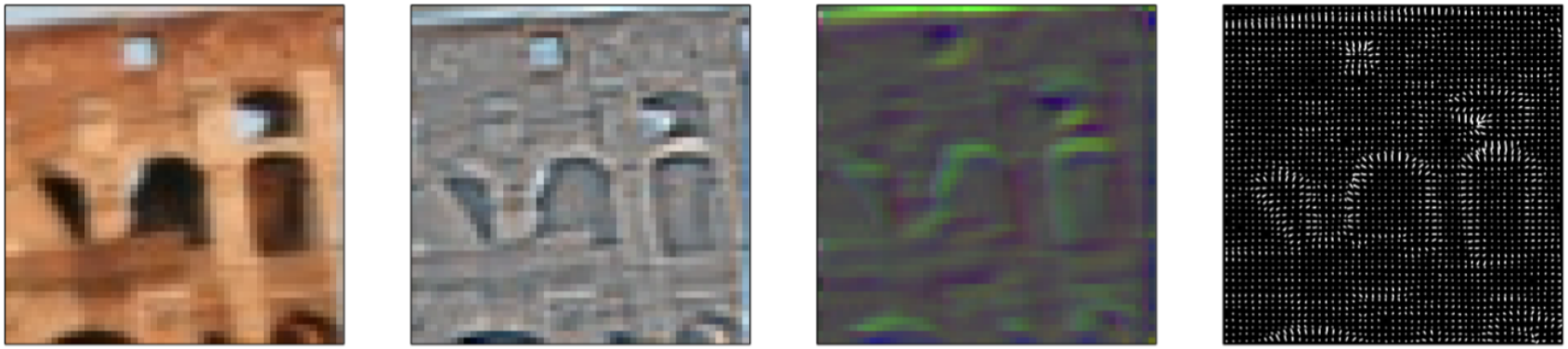


Figure: Vizualization of the possible CKN inputs: raw patch (left); patches where individual sub-patch means have been subtracted (middle-left); CKN-white, similar to the previous but with whitening (middle-right) and CKN-grad (right). Sub-patches were extracted with a size of 3x3; reconstructed images are the pixelwise means of the modified subpatches (up to normalization to fit the 0-256 range).

Experiments: image retrieval datasets

Holidays

The Holidays dataset contains 500 different scenes or objects, for which 1491 views are available. 500 images are excluded from the dataset and serve as queries.

Oxford

The Oxford dataset consists of 5000 images of Oxford landmarks. 11 locations of the city are selected as queries and 5 views per location is available. The standard benchmarking, which we use, involves a cropping of the bounding-box of the region of interest in the query view, followed by retrieval.

UKB

The University of Kentucky Benchmark (UKB) contains 10,200 photos. They consists of 4 different views of the same object, under radical viewpoint changes. All images are used as queries in turn, and the standard measure is the mean number of true positives returned in the four first retrieved images.

Results on image retrieval

	Oxford	UKB	Holidays
SIFT	43.7	3.44	64.0
AlexNet	34.3	3.74	79.3
PhilippNet	43.6	3.67	74.7
CKN	**49.8**	**3.80**	**79.3**

Supervised approaches, trained on Rome:
- AlexNet
- PhilippNet

Un-supervised approaches:
- CKN
- SIFT

CKN: Descriptor based on two-layer Convolutional Kernel Nets

Results in Mean Average Precision on the benchmark datasets *Oxford*, *UKB*, and *Holidays*.

Motion estimation

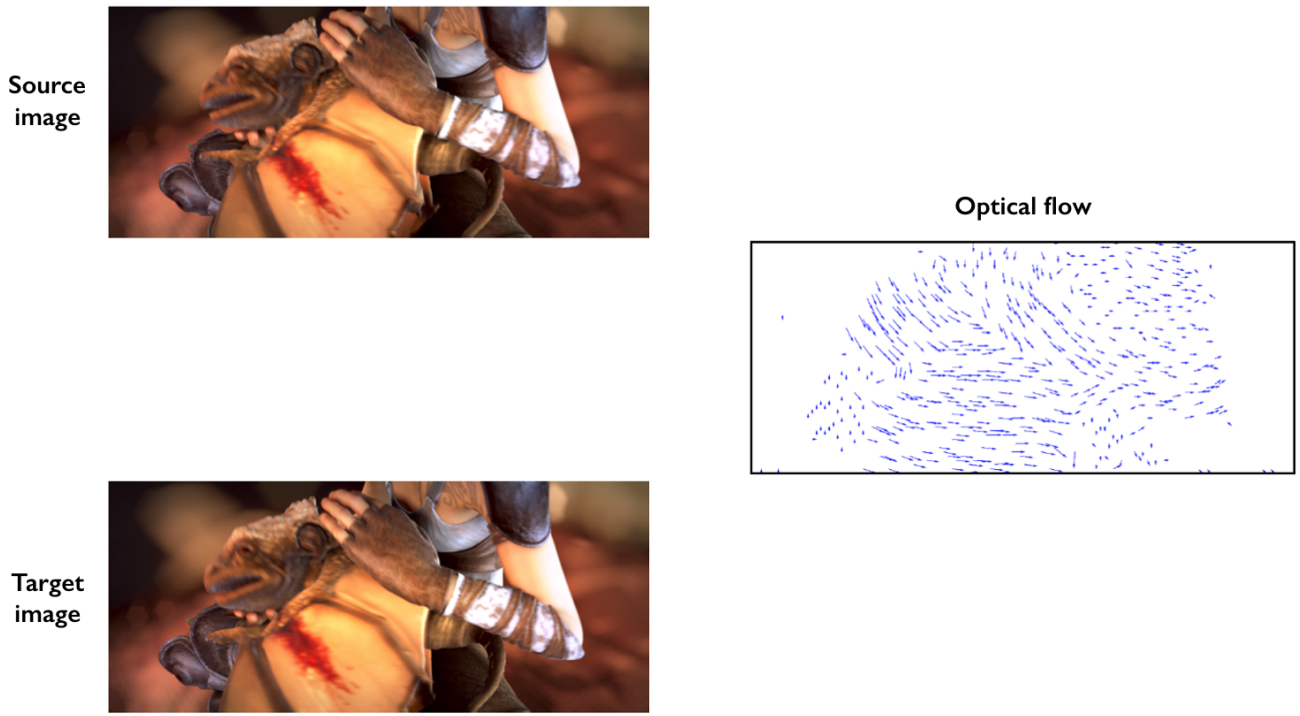
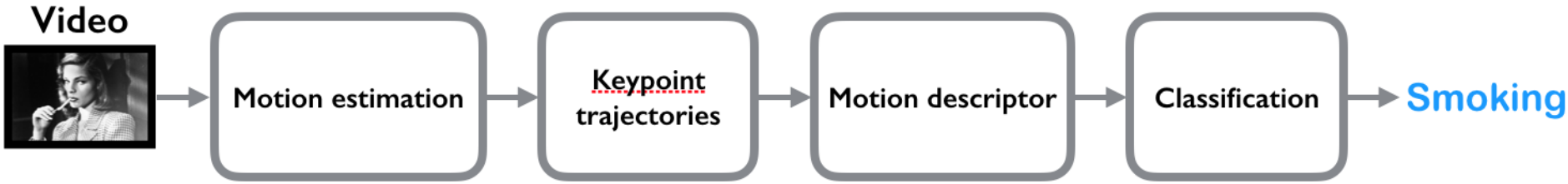


Figure: Motion estimation or optical flow estimation (MPI-Sintel dataset).

Motion estimation for action recognition

Motion estimation for action recognition



Motion estimation is key for action recognition.

Motion estimation for action localization

Estimation of fast motion (large displacement)

Methods based on convolutional matching, **DeepMatching** and later versions, settled a **new state-of-the-art** for fast motion optical flow estimation on MPI-Sintel and Kitti benchmarks.

Method	AEE	AEE-occ	s0-10	s10-40	s40+	Time
EpicFlow	**6.285**	**32.564**	**1.135**	**3.727**	**38.021**	**16.4s**
TF+OFM	6.727	33.929	1.512	3.765	39.761	∼400s
DeepFlow	7.212	38.781	1.284	4.107	44.118	19s
S2D-Matching	7.872	40.093	1.172	4.695	48.782	∼2000s
Classic+NLP	8.291	40.925	1.208	5.090	51.162	∼800s
MDP-Flow2	8.445	43.430	1.420	5.449	50.507	709s
NLTGV-SC	8.746	42.242	1.587	4.780	53.860	
LDOF	9.116	42.344	1.485	4.839	57.296	30s

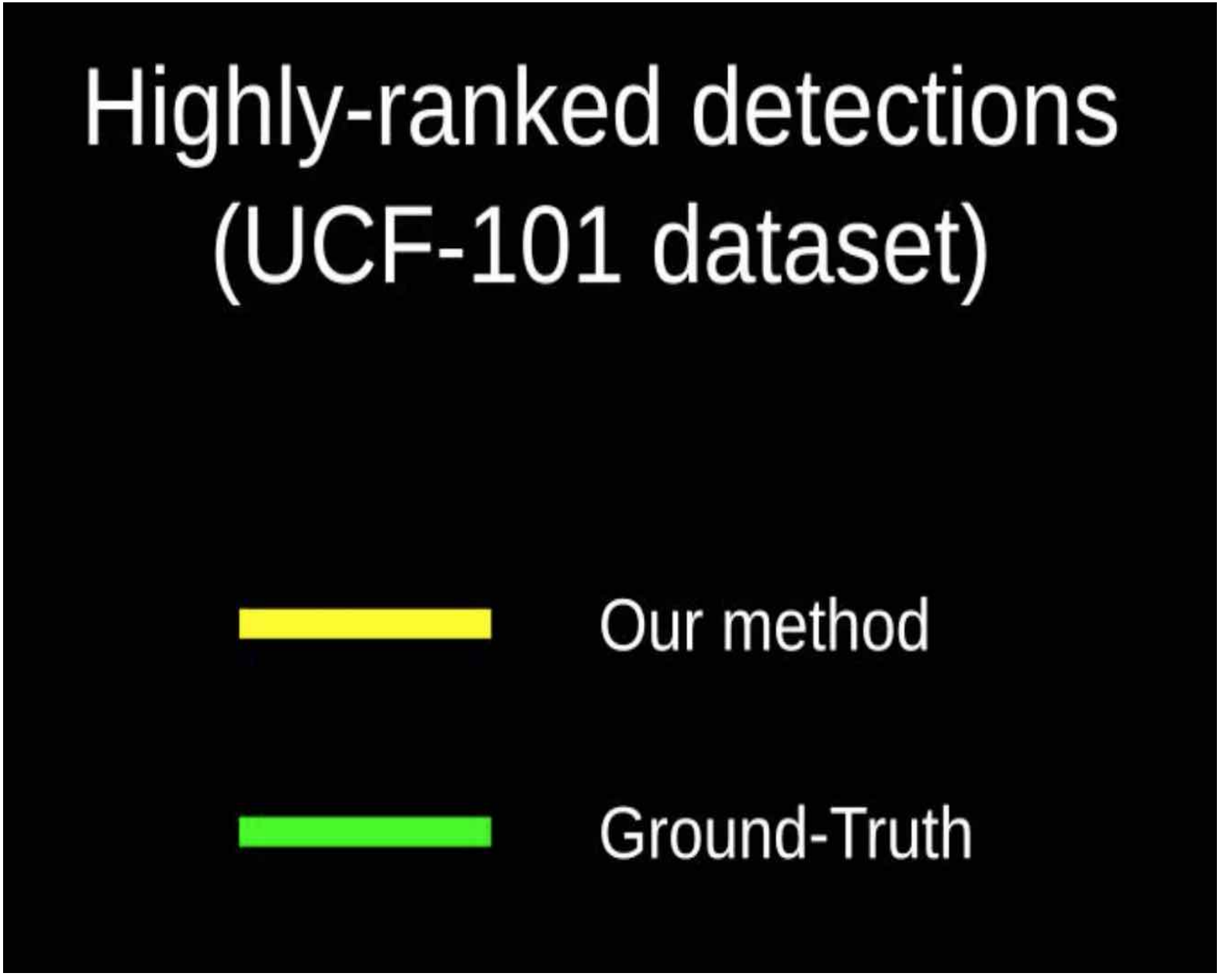
Results for optical flow estimation on the benchmark dataset MPI-Sintel.

Motion estimation for action localization

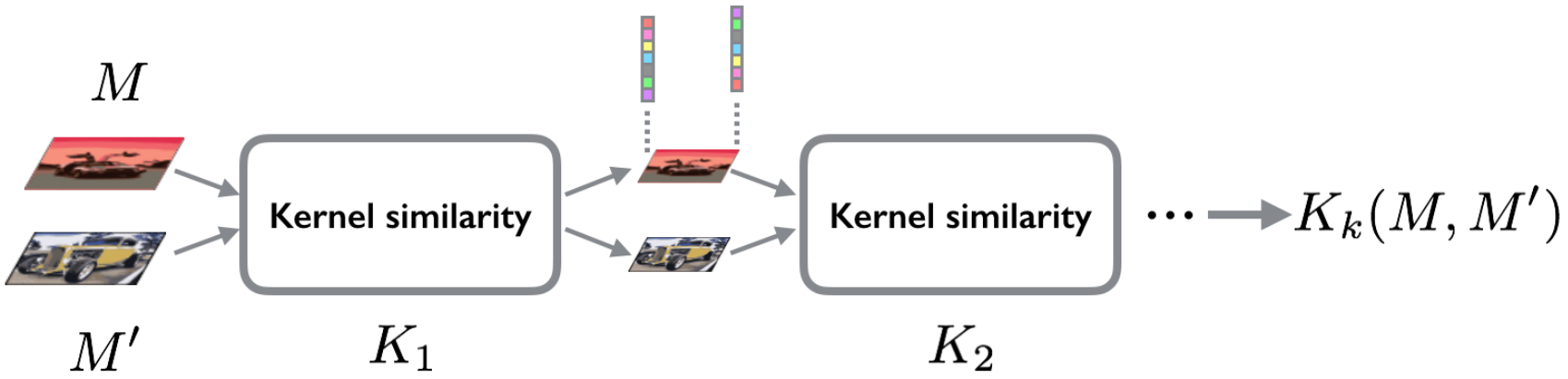
Spatio-temporal action localization

	J-HMDB	UCF-101
Gkioxari & Malik Yu and Yuan 2015	53.3	42.8
Ours	**60.7**	**54.3**

Results in mean average precision (mAP) for spatio-temporal localization of actions on the benchmark datasets J-HMDB and UCF-101.


Highly-ranked detections
(UCF-101 dataset)
Our method
Ground-Truth

Towards unsupervised training of deep nets



- approach to design **similarity** between images
- simpler and trainable in an **unsupervised manner**
- alternative to **convolutional neural nets trained with supervision**

Papers

Convolutional Patch Representations for Image Retrieval: an Unsupervised Approach
Mattis Paulin, Julien Mairal, Matthijs Douze, Zaid Harchaoui, Florent Perronnin, Cordelia Schmid
International Jounral on Computer Vision (IJCV), submitted, 2016

Local Convolutional Features with Unsupervised Training for Image Retrieval
Mattis Paulin, Matthijs Douze, Zaid Harchaoui, Julien Mairal, Florent Perronnin, Cordelia Schmid
International Conference on Computer Vision (ICCV), 2015

Convolutional Kernel Networks
Julien Mairal, Piotr Koniusz, Zaid Harchaoui, Cordelia Schmid
Advances in Neural Information Processing Systems (NIPS), 2014

The code for CKNs is available
`http://ckn.gforge.inria.fr`